Lab2
Scan Chain Insertion and ATPG Using DFTADVISOR and FASTSCAN

Pro: Chia-Tso Chao

TA: Yu-Teng Nien

2017-05-14

Outline

- □ Introduction
- □ Dftadvisor
- □ Fastscan
- □ Mix Flow
- □ Lab

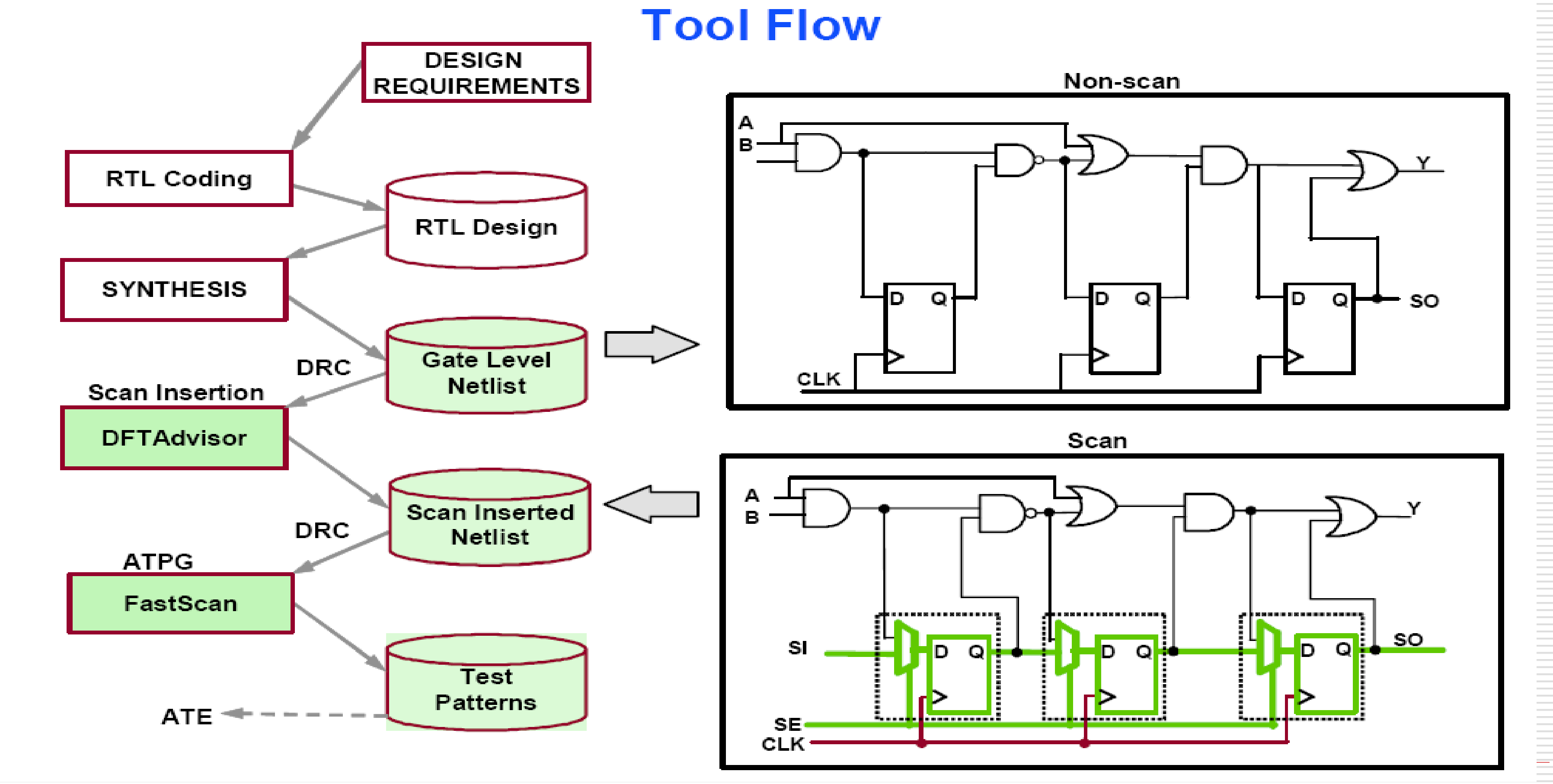
Outline

- □ Introduction
- □ DFTADVISOR
- □ FASTSCAN
- □ Mix Flow
- □ Lab

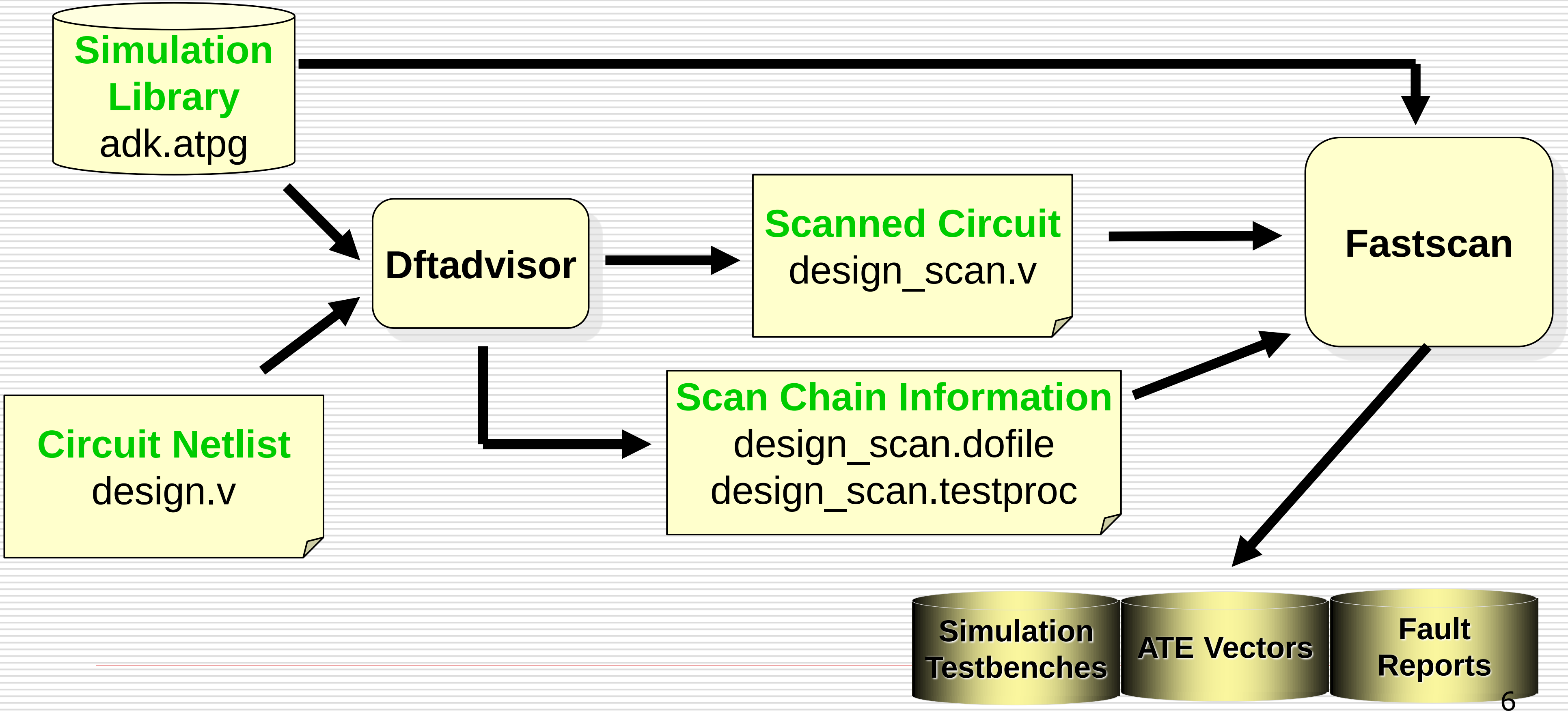
Introduction

- This lab focuses on ATPG results from different tools:
  Mentor Graphic and Synopsys.
- Dftadvisor is used to insert scan chain (basically replace FF with scan FF).
- Fastscan is used to do ATPG and fault simulation.

Insert Scan and ATPG Flow



Input/Output Files



Simulation Library
adk.atpg

Circuit Netlist
design.v

Dftadvisor

Scanned Circuit
design_scan.v

Scan Chain Information
design_scan.dofile
design_scan.testproc

Fastscan

Simulation Testbenches

ATE Vectors

Fault Reports

Outline

- □ Introduction
- □ DFTADVISOR
- □ FASTSCAN
- □ Mix Flow
- □ Lab

Invoke DFTADVISOR

- Read in Verilog source file and assign library file.
  - $ dftadvisor pre_norm_noscan.v -verilog -lib l90sprvt.atpg -nogui

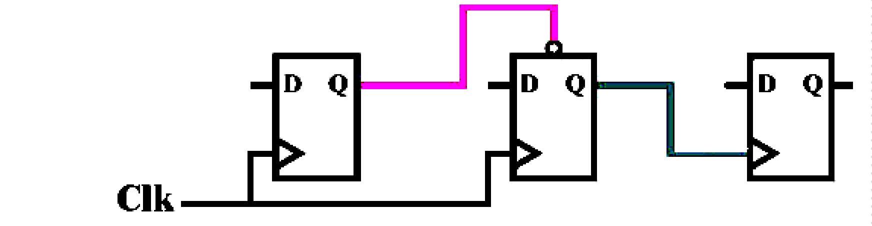
Specify Clock

- Clocks are primary input signals that asynchronously change the state of sequential logic elements.
  - SETUP> add_clocks 0 clk

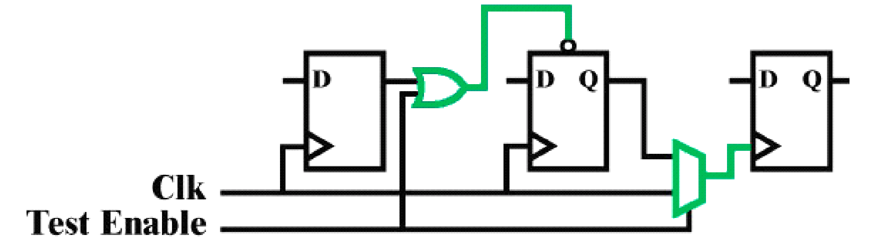
specify state which cannot affect output

Setup Test Logic Configuration

- □ Set scan methodology.
  - ■ Mux_scan : mux-DFF
  - ■ Lssd : level sensitive DFF
  - ■ Clocked_scan : clocked-signal
    - □ SETUP> set_scan_type m
- □ Test logic options-- make clock lines controllable to get a scannable design.
  - ■ SETUP> set_test_logic
    -clock on -reset on



Non-scannable



Scannable

Scan Insertion

- Enter DFT mode.
  - SETUP> set_system_mode dft
- Perform scan.
  - DFT> run
- Report detailed statistics report.
  - DFT> report_statistics

DRC Rules

- General Rules
- Procedure Rules
- Scan Chain Trace Rules
- Scan Cell Data Rules
- Clock Rules
- Ram Rules
- BIST Rules
- EDT Rules
- Timing Rules

Insert Scan Chain and View Report

- Set # of scan-chains to insert and do so.
  - DFT> insert_test_logic -number 10
- Report scan-chain information.
  - DFT> report_scan_chains
  - DFT> report_test_logic

Output Scanned Design for ATPG

- Write out files and exit DFTADVISOR
  - DFT> write netlist pre_norm_scan.v -verilog -replace
  - DFT> write_atpg_setup pre_norm_scan -replace
  - DFT> exit

.dofile : setup information
.testproc : procedure file

Outline

- □ Introduction
- □ DFTADVISOR
- □ FASTSCAN
- □ Mix Flow
- □ Lab

Invoke Fastscan

- specify scannable Verilog file and library file.
  - $ fastscan pre_norm_scan.v -verilog -lib l90sprvt.atpg -nogui

Read Setup Information

- Read setup information from Dftadvisor.
  - SETUP> dofile pre_norm_scan.dofile
- Enter atpg mode.
  - SETUP> set_system_mode atpg
- Setup fault type: stuck, IDDQ, toggle, transition.
  - ATPG> set_fault_type stuck

Generate Patterns

- Uses '-auto' option to
  - Analyze the design.
  - Suggest the best settings possible to generate the most compact patterns with the highest coverage within the lowest time.
  - ATPG> create_patterns -auto

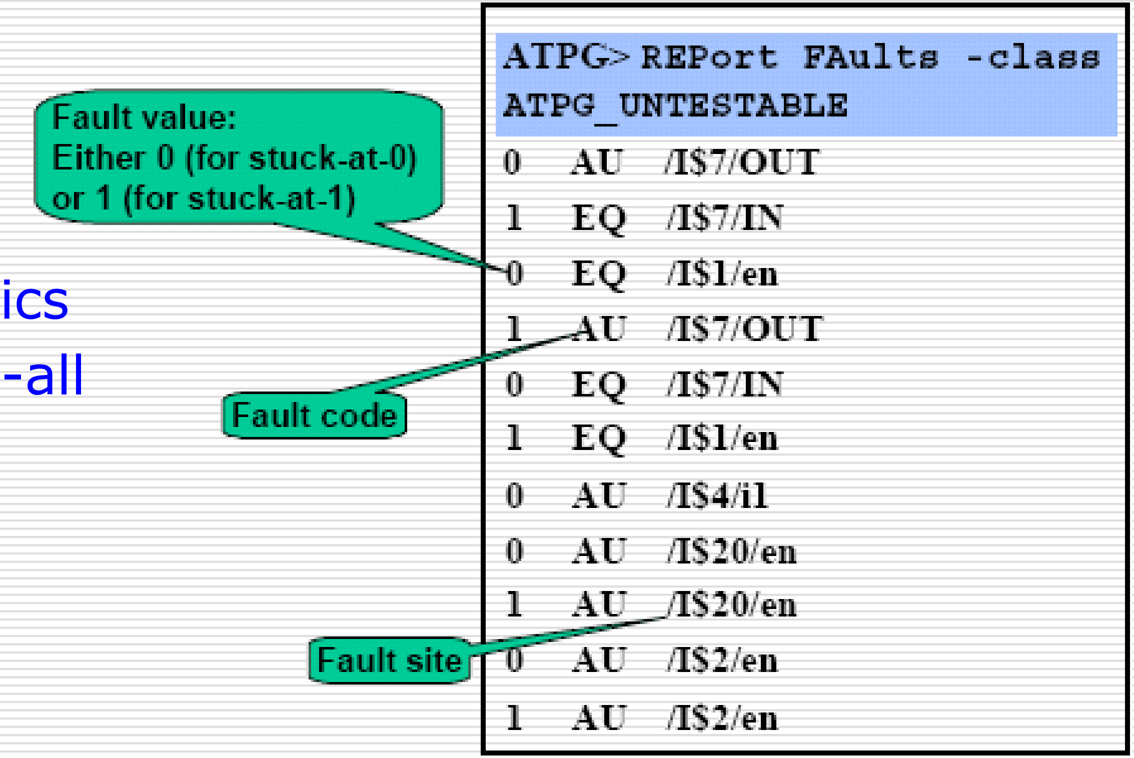
Generate Patterns

- Without ‘auto’ option, you can specify your own configurations using these commands :
  - set_atpg_limits -cpu_seconds [integer] -test_coverage [real] -pattern_count [integer]
  - set_atpg_compression on –abort_limit [integer]
  - identify_redundant_faults

  - ATPG> create_patterns

View Report

- Report simulation result and faults.
  - ATPG> report_statistics
  - ATPG> report_faults -all



```
ATPG> REPort FAults -class
ATPG_UNTESTABLE
0   AU   /I$7/OUT
1   EQ   /I$7/IN
0   EQ   /I$1/en
1   AU   /I$7/OUT
0   EQ   /I$7/IN
1   EQ   /I$1/en
0   AU   /I$4/i1
0   AU   /I$20/en
1   AU   /I$20/en
0   AU   /I$2/en
1   AU   /I$2/en
```

Fault value:
Either 0 (for stuck-at-0)
or 1 (for stuck-at-1)

Fault code

Fault site

Result(1/2)

```
□  //  Simulation performed for #gates = 27550  #faults = 110115
□  //  system mode = ATPG    pattern source = internal patterns
□  //  ----------------------------------------------------------------------
□  //  #patterns  test      #faults  #faults    # eff.   # test     process    RE/AU/abort
□  //  simulated  coverage  in list  detected   patterns  patterns    CPU time
□  //  deterministic ATPG invoked with comb/seq abort limit = 300/100
□  //   ---       ------     ---      ---      ---      ---      0.10 sec    224/0/0
□  //   32         82.90%    20861    89030     32       32        0.15 sec
□  //   ---       ------     ---      ---      ---      ---      0.25 sec    1423/0/0
□  //   64         91.30%    10501     9161     32       64        0.26 sec
□  //   ---       ------     ---      ---      ---      ---      0.40 sec    2342/0/0
□  //   96         95.67%     5187     4395     32       96        0.41 sec
□  //   ---       ------     ---      ---      ---      ---      0.59 sec    3192/0/0
□  //   128        98.43%     1887     2450     32      128        0.60 sec
□  //   ---       ------     ---      ---      ---      ---      0.67 sec    3609/0/0
□  //   160        99.22%      954      516     10      138        0.68 sec
□  //   ---       ------     ---      ---      ---      ---      0.68 sec    3609/0/0
□  //   192        99.69%      357      597     32      170        0.69 sec
□  //   ---       ------     ---      ---      ---      ---      0.69 sec    3609/0/0
□  //   224        99.98%       24      333     32      202        0.70 sec
□  //   ---       ------     ---      ---      ---      ---      0.70 sec    3609/0/0
□  //   256       100.00%        0       24      3      205        0.70 sec
```

Result(2/2)

```
Statistics report
-------------------------------------------
Fault Classes                #faults
                         (total)
----------------------   --------------
  FU (full)                 6078
  --------------------   --------------
  UO (unobserved)            9 ( 0.15%)
  DS (det_simulation)    5393 (88.73%)
  DI (det_implication)    540 ( 8.88%)
  UU (unused)             16 ( 0.26%)
  RE (redundant)          118 ( 1.94%)
  AU (atpg_untestable)      2 ( 0.03%)
--------------------------------------
Coverage
  --------------------
  test_coverage              99.81%
  fault_coverage             97.61%
  atpg_effectiveness          99.85%
--------------------------------------
#test_patterns                  180
#simulated_patterns               224
CPU_time (secs)                  3.7
--------------------------------------
```

Fault Types(1/3)

- AU : Atpg_untestable
  - Due to pin constraint or insufficient sequential depth cause testable faults become atpg_untestable
- FU : Full
- TE : Testable
- DT : Detected

Fault Types(2/3)

- UT: Untestable
  - Faults which no pattern can exist to either detect or possible-detect them, such as unused pins
- UD : Undetected
  - Faults cannot be proven untestable or ATPG_untestable
- RE : Redundant

Fault Types(3/3)

- UU: usused
  - All faults unconnected to any circuit observation point
- BL: blocked
  - Faults which logic blocks all paths to an observation point
- TI:tied
  - Point of the fault value is always same (and-gate with complementary inputs)

Test Coverage Formula Comparation

- □ Tmax

$$\text{test \_ coverage} = \frac{DT + (\overset{\overset{\text{possible detected}}{\uparrow}}{PT} * \overset{\text{default 50\%}}{\text{posdet\_credit}})}{all\_faults - (UD + AU * \overset{\text{default 0}}{\text{au\_credit}})}$$

- □ Fastscan

$$\text{test\_cover age} = \frac{DT + (PT * \overset{\text{default 50\%}}{\text{posdet\_credit}})}{\text{testable}} * 100$$

$$\text{fault\_cove rage} = \frac{DT + (PT * \text{posdet\_credit})}{\text{full}} * 100$$

$$\text{ATPG\_effec tiveness} = \frac{DT + UT + AU + PU + (PT * \text{posdet\_credit})}{\text{full}} * 100$$

Testable=DT+PT+AU+UD Untestable=UU+TI+BL+RE

Save Patterns

- Save patterns just generated. The response of ATE saved in s38584_seq_ate.pat
- Various format including binwgl, ctl2005, stil2005, stil999, Verilog, VHDL, wgl, zycad, tstl2, utic.

  -
    ```
    ATPG> save patterns pre_norm_scan.pat
                        -verilog –proc –replace
    ```

  -
    ```
    ATPG> save patterns pre_norm_scan_tstl2.pat
                        -TSTL2 –rep
    ```

  -
    ```
    ATPG> exit
    ```

Toshiba Standard Tester Interface Language 2

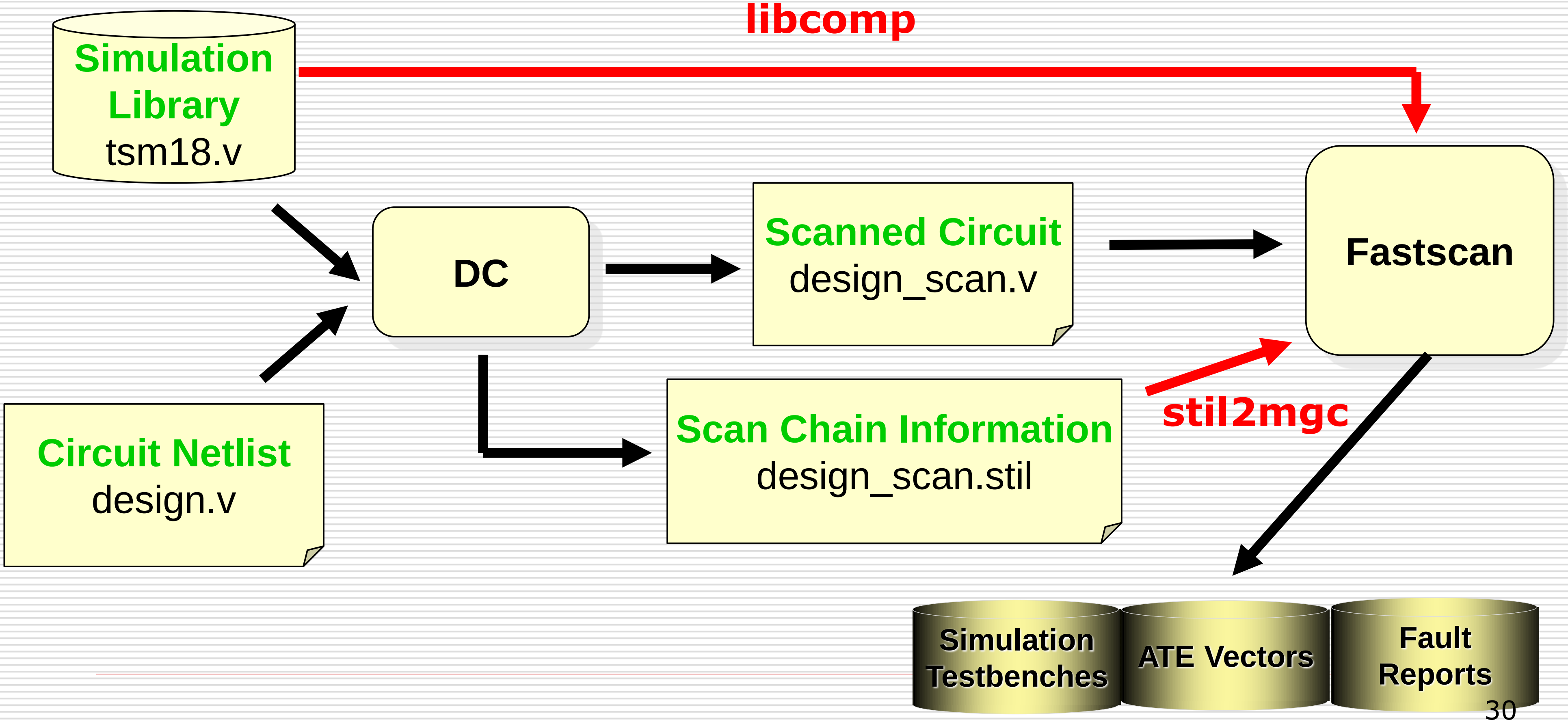
Outline

- □ Introduction
- □ DFTADVISOR
- □ FASTSCAN
- □ Mix Flow
- □ Lab

Testing Flow

- Synopsys Design Compiler is better in mapping from RTL code to gate-level circuit.
- Some cases in industry field use Design Compiler to synthesize RTL code and use Fastscan to perform ATPG and fault simulation.

Input/Output Files



Input Files Required from Design Compiler's Flow

- Library file needs to be converted.
  - From .v to .atpg
- The detail information about scanned circuit need to be converted.
  - From .stil to .dofile
- Scanned code.

Convert STIL File

- Using command 'stil2mgc' to convert STIL file into dofile and test procedure file for Fastscan.
  - $ stil2mgc pre_norm_scan.stil
    - It will generate pre_norm_scan.stil.do and pre_norm_scan.stil.proc

Performing ATPG using FASTSCAN

- Read scanned circuit and library from design compiler to perform ATPG.
  - $ fastscan pre_norm_scan.v -verilog -lib l90sprvt.atpg -nogui
  - SETUP> dofile pre_norm_scan.stil.do
  - SETUP> set_system_mode atpg
  - ATPG> create_patterns -auto
  - ATPG> report_statistics

Outline

- □ Introduction
- □ DFTADVISOR
- □ FASTSCAN
- □ Mix Flow
- □ Lab

Lab Goal

- Compare test coverage, number of patterns and run time during ATPG using flows in DC+TMAX and DFTA+FS and DC+FS.

- You need to run in circuits pre_norm.v, and show the results like next slide.

Result

	Total Faults	Test Coverage	# of Patterns	Run time
DC + TMAX	71298	100%	138	0.83s
DFTA + FS	122072	100%	201	0.66s
DC + FS	75208	100%	203	0.51s

References

- Mentor Graphic User Guide
- Synopsys TetraMax User Guide